# CONTENTS

**Chapter 1:** Understanding Live Agent REST Resources ........................................... 1

**Chapter 2:** Making Requests with your Live Agent API Endpoint ............................ 2

**Chapter 3:** Request Headers ...................................................................................... 3

**Chapter 4:** Your Message Long Polling Loop .............................................................. 4

**Chapter 5:** Live Agent REST API Resources ............................................................. 5
Create a Live Agent Session ......................................................................................... 5
SessionId ...................................................................................................................... 5
Create a Chat Visitor Session ...................................................................................... 6
ChasitorInit .................................................................................................................. 6
ResyncSession ............................................................................................................ 7
ChasitorResyncState ................................................................................................... 8
Monitor Chat Activity .................................................................................................. 9
ChasitorNotTyping ....................................................................................................... 9
ChasitorSneakPeek ...................................................................................................... 10
ChasitorTyping ........................................................................................................... 10
ChatEnd ....................................................................................................................... 11
ChatMessage ................................................................................................................ 12
CustomEvent ............................................................................................................... 12
Messages ..................................................................................................................... 13
MultiNoun .................................................................................................................... 19
Customize the Chat Visitors’ Experience ..................................................................... 20
Settings ....................................................................................................................... 20
Availability ................................................................................................................... 21
Breadcrumb ................................................................................................................ 22
VisitorId ....................................................................................................................... 22

**Chapter 6:** Request Bodies for Live Agent REST API .................................................. 24

**Chapter 7:** Response Bodies for Live Agent REST API ............................................... 28

**Chapter 8:** Live Agent REST API Data Types ............................................................ 41

**Chapter 9:** Status Codes and Error Responses .......................................................... 48

Index ............................................................................................................................ 49
The Live Agent REST API allows you to customize and extend the functionality of Live Agent chat windows.

With the Live Agent REST API, you no longer have to rely on Visualforce to develop customized chat windows. The Live Agent REST API resources let your developers to extend the functionality of chat windows beyond simple HTML and JavaScript environments, allowing them to develop chat windows that merge seamlessly into your company’s own applications. With the REST resources in this guide, your developers can produce chat windows that go beyond your website: For example, you can build custom windows within your company’s iOS, Android, or other web applications.
To make requests to the Live Agent server, you must use your Live Agent API endpoint.

Your Live Agent API endpoint is a unique URL that lets you access data from your organization’s Live Agent sessions.

To find your organization’s Live Agent API endpoint, from Setup, enter “Live Agent Settings” in the Quick Find box, then select Live Agent Settings.

Replace the hostname variable with your API endpoint in the URL for each new request.
Each Live Agent REST API resource requires one or more headers to make a request.

Not all resources require all of the available request headers. Each resource indicates which headers are required to make a request.

The following headers are available:

<table>
<thead>
<tr>
<th>Header Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-LIVEAGENT-API-VERSION</td>
<td>The Salesforce API version for the request.</td>
</tr>
<tr>
<td>X-LIVEAGENT-AFFINITY</td>
<td>The system-generated ID used to identify the Live Agent session on the Live Agent servers. This affinity token is included in the response body of the SessionId request.</td>
</tr>
<tr>
<td>X-LIVEAGENT-SESSION-KEY</td>
<td>The unique ID associated with your Live Agent session.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Your session key shouldn’t be shared or sent over insecure channels, as it allows access to potentially sensitive chat information.</td>
</tr>
<tr>
<td>X-LIVEAGENT-SEQUENCE</td>
<td>The sequence of messages you have sent to the Live Agent server to help the Live Agent server avoid processing duplicate messages. This number should be increased by one with every new request.</td>
</tr>
</tbody>
</table>
Message long polling notifies you of events that occur on the Live Agent server for your Live Agent session. When you start a request, all pending messages will be immediately delivered to your session. If there are no pending messages, the connection to the server will remain open. The connection will return messages continuously as they are received on the server.

If your session expires, you will receive a 200 ("OK") response code and a resource that contains an array of the remaining messages. If no messages were received, you will receive a 204 ("No Content") response code.

When you receive a 200 ("OK") or 204 ("No Content") response code, immediately perform another Messages request to continue to retrieve messages that are registered on the Live Agent server.

⚠️ Warning: If you don’t make another Messages request to continue the messaging loop, your session will end after a system timeout on the Live Agent server.

If you don’t receive a response within the number of seconds indicated by the clientPollTimeout property in your SessionId request, your network connection to the server is likely experiencing an error, so you should terminate the request.

To initiate a long polling loop, perform a Messages request.

SEE ALSO:
- Messages
- SessionId
- Status Codes and Error Responses
CHAPTER 5  Live Agent REST API Resources

To perform a POST or GET request, create and send an HTTP request with the appropriate parameters or request body. The Live Agent REST API requests let you begin new chat sessions between agents and chat visitors and monitor the chat activity that occurs.

IN THIS SECTION:

Create a Live Agent Session
To create a new Live Agent session, you must call the SessionId request.

Create a Chat Visitor Session
To create or reestablish a chat visitor session using the Live Agent REST API, you must make certain requests.

Monitor Chat Activity
Live Agent requests indicate when certain activities occurred during a chat session.

Customize the Chat Visitors' Experience
With the Live Agent visitor REST API resources, you can establish your chat visitors’ experience with Live Agent in custom mobile applications.

Create a Live Agent Session

To create a new Live Agent session, you must call the SessionId request.

IN THIS SECTION:

SessionId
Establishes a new Live Agent session. The SessionId request is required as the first request to create every new Live Agent session.

Syntax

**URI**

https://hostname/chat/rest/System/SessionId

**Available since release**
This resource is available in API versions 29.0 and later.

**Formats**

JSON

**HTTP methods**

GET
Create a Chat Visitor Session

To create or reestablish a chat visitor session using the Live Agent REST API, you must make certain requests.

IN THIS SECTION:
- ChasitorInit
- ResyncSession
- ChasitorResyncState

ChasitorInit

Initiates a new chat visitor session. The ChasitorInit request is always required as the first POST request in a new chat session.

Syntax

URI

https://hostname/chat/rest/Chasitor/ChasitorInit

Available since release

This resource is available in API versions 29.0 and later.

Formats

JSON

HTTP methods

POST

Request headers

- X-LIVEAGENT-API-VERSION
- X-LIVEAGENT-AFFINITY
ResyncSession

Reestablishes a customer's chat session on a new server if the session is interrupted and the original server is unavailable.

This request should only be made if you receive a 503 response status code, indicating that the affinity token has changed for your Live Agent session. When you receive a 503 response status code, you must cancel any existing inbound or outbound requests.

The data in outbound requests will be temporarily stored and resent once the session is reestablished. Upon receiving the response for the ResyncSession request, you can start polling for messages if the isValid response property is true.

The first response will be a ChasitorSessionData message containing the data from the previous session that will be restored once the session is reestablished. After receiving that message, you can proceed to send the existing messages that were cancelled upon receiving the 503 response status code.

Syntax

URI

https://hostname/chat/rest/System/ResyncSession

Available since release
This resource is available in API versions 29.0 and later.

Formats
JSON

HTTP methods
GET

Request headers
X-LIVEAGENT-API-VERSION
X-LIVEAGENT-AFFINITY
X-LIVEAGENT-SESSION-KEY

Request parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SessionId</td>
<td>String</td>
<td>The chat session ID from the SessionId request.</td>
</tr>
</tbody>
</table>
Query parameters
None

Request body
None

Response body
ResyncSession response

SEE ALSO:
Status Codes and Error Responses
ChasitorSessionData
ChasitorResyncState
Status Codes and Error Responses

ChasitorResyncState
Reestablishes the chat visitor’s state, including the details of the chat, after a ResyncSession request is completed.

Syntax

URI
https://hostname/chat/rest/Chasitor/ChasitorResyncState

Available since release
This resource is available in API versions 29.0 and later.

Formats
JSON

HTTP methods
POST

Request headers
X-LIVEAGENT-API-VERSION
X-LIVEAGENT-AFFINITY
X-LIVEAGENT-SESSION-KEY

Request parameters
None

Query parameters
None

Request body
ChasitorResyncState request

Response body
None

SEE ALSO:
ResyncSession
Monitor Chat Activity

Live Agent requests indicate when certain activities occurred during a chat session.

IN THIS SECTION:

ChasitorNotTyping
Indicates that the chat visitor is not typing in the chat window.

ChasitorSneakPeek
Provides a chat visitor’s message that was viewable through Sneak Peek.

ChasitorTyping
Indicates that a chat visitor is typing a message in the chat window.

ChatEnd
Indicates that a chat visitor has ended the chat.

ChatMessage
Returns the body of the chat message sent by the chat visitor.

CustomEvent
Indicates a custom event was sent from the chat visitor during the chat.

Messages
Returns all messages that were sent between agents and chat visitors during a chat session.

MultiNoun
Batches multiple POST requests together if you’re sending multiple messages at the same time.

ChasitorNotTyping
Indicates that the chat visitor is not typing in the chat window.

Syntax

URI
https://hostname/chat/rest/Chasitor/ChasitorNotTyping

Available since release
This resource is available in API versions 29.0 and later.

Formats
JSON

HTTP methods
POST

Request headers
X-LIVEAGENT-API-VERSION
X-LIVEAGENT-AFFINITY
X-LIVEAGENT-SESSION-KEY
X-LIVEAGENT-SEQUENCE
ChasitorSneakPeek

Provides a chat visitor's message that was viewable through Sneak Peek.

Syntax

URI

https://hostname/chat/rest/Chasitor/ChasitorSneakPeek

Available since release

This resource is available in API versions 29.0 and later.

Formats

JSON

HTTP methods

POST

Request headers

X-LIVEAGENT-API-VERSION
X-LIVEAGENT-AFFINITY
X-LIVEAGENT-SESSION-KEY
X-LIVEAGENT-SEQUENCE

Request parameters

None

Query parameters

None

Request body

ChasitorSneakPeek request

Response body

None

ChasitorTyping

Indicates that a chat visitor is typing a message in the chat window.
ChatEnd

Indicates that a chat visitor has ended the chat.

Syntax

URI
https://hostname/chat/rest/Chasitor/ChatEnd

Available since release
This resource is available in API versions 29.0 and later.

Formats
JSON

HTTP methods
POST

Request headers
X-LIVEAGENT-API-VERSION
X-LIVEAGENT-AFFINITY
X-LIVEAGENT-SESSION-KEY
X-LIVEAGENT-SEQUENCE

Request parameters
None

Query parameters
None

Request body
None

Response body
None
ChatEndReason

—

Include the ChatEndReason parameter in the request body of your request to specify the reason that the chat ended. This parameter is required. For example: {reason: "client"}.

Response properties

attachedRecords—Includes attached record IDs. You can use this Visualforce component to display the attached record IDs in the post-chat page: <apex:outputText value="{!$CurrentPage.parameters.attachedRecords}"/>

ChatMessage

Returns the body of the chat message sent by the chat visitor.

Syntax

URI

https://hostname/chat/rest/Chasitor/ChatMessage

Available since release

This resource is available in API versions 29.0 and later.

Formats

JSON

HTTP methods

POST

Request headers

X-LIVEAGENT-API-VERSION
X-LIVEAGENT-AFFINITY
X-LIVEAGENT-SESSION-KEY
X-LIVEAGENT-SEQUENCE

Request parameters

None

Query parameters

None

Request body

ChatMessage request

Response body

None

CustomEvent

Indicates a custom event was sent from the chat visitor during the chat.
Syntax

URI
https://hostname/chat/rest/Chasitor/CustomEvent

Available since release
This resource is available in API versions 29.0 and later.

Formats
JSON

HTTP methods
POST

Request headers
X-LIVEAGENT-API-VERSION
X-LIVEAGENT-AFFINITY
X-LIVEAGENT-SESSION-KEY
X-LIVEAGENT-SEQUENCE

Request parameters
None

Query parameters
None

Request body
CustomEvent request

Response body
None

Messages

Returns all messages that were sent between agents and chat visitors during a chat session.

For a complete list of responses for the Messages resource, see Live Agent REST API Messages Response Objects.

Syntax

URI
https://hostname/chat/rest/System/Messages

Available since release
This resource is available in API versions 29.0 and later.

Formats
JSON

HTTP methods
GET

Request headers
X-LIVEAGENT-API-VERSION
X-LIVEAGENT-AFFINITY
X-LIVEAGENT-SESSION-KEY

Request body

Response body
None
Request parameters
None

Query parameters
ack—The ack query parameter is a sequencing mechanism that allows you to poll for messages on the Live Agent server. The first time you make the Messages request, the ack parameter is set to -1; this value increases by 1 with each new Messages request you send. To guarantee that you receive the messages from the server in the correct order, you must update the ack in the next request you make to the value you receive in the preceding response.

Request body
None

Response body

IN THIS SECTION:

Live Agent REST API Messages Response Objects

The Messages request returns an array of objects that represent all the events that occurred during an agent’s chat with a chat customer.

SEE ALSO:

Your Message Long Polling Loop

Live Agent REST API Messages Response Objects

The Messages request returns an array of objects that represent all the events that occurred during an agent’s chat with a chat customer.

This request can return several subtypes with unique response bodies, depending on the events that occurred within the chat.

Here is an example of the structure of a Messages response array:

```
{
  "messages":{
    "type":"array",
    "description":"The messages sent over the course of a chat.",
    "items":{
      "name":"Message",
      "type":"object",
      "properties": {
        "type": { 
          "type":"string",
          "description":"The type of message that was received.",
          "required":true,
          "version":29.0
        },
        "message": { 
          "type":"object",
          "description":"A placeholder object for the message that was received. Can return any of the responses available for the Messages request.",
          "required":true,
          "version":29.0
        }
      }
    }
  }
}
```
IN THIS SECTION:

- **AgentDisconnect**
  Indicates that the agent has been disconnected from the chat.

- **AgentNotTyping**
  Indicates that the agent is not typing a message to the chat visitor.

- **AgentTyping**
  Indicates that the agent is typing a message to the chat visitor.

- **ChatSessionData**
  Returns the current chat session data for the chat visitor. This request is used to restore the session data for a chat visitor's chat session after a ResyncSession request is sent.

- **ChatEnded**
  Indicates that the chat has ended.

- **ChatEstablished**
  Indicates that an agent has accepted a chat request and is engaged in a chat with a visitor.

- **ChatMessage**
  Indicates a new chat message has been sent from an agent to a chat visitor.

- **ChatRequestFail**
  Indicates that the chat request was not successful.

- **ChatRequestSuccess**
  Indicates that the chat request was successful and routed to available agents.

- **ChatTransferred**
  Indicates the chat was transferred from one agent to another.

- **CustomEvent**
  Indicates a custom event was sent from an agent to a chat visitor during a chat.

- **NewVisitorBreadcrumb**
  Indicates the URL of the Web page the chat visitor is currently viewing.

- **QueueUpdate**
  Indicates the new position of the chat visitor in the chat queue when the visitor's position in the queue changes.
AgentDisconnect
Indicates that the agent has been disconnected from the chat.

Note: Though the agent has been disconnected from the chat, the chat session is still active on the server. A new agent may accept the chat request and continue the chat.

Syntax
Available since release
This resource is available in API versions 29.0 and later.
Response body
None
Response properties
None

AgentNotTyping
Indicates that the agent is not typing a message to the chat visitor.

Syntax
Available since release
This resource is available in API versions 29.0 and later.
Response body
None
Response properties
None

AgentTyping
Indicates that the agent is typing a message to the chat visitor.

Syntax
Available since release
This resource is available in API versions 29.0 and later.
Response body
None
Response properties
None

ChasitorSessionData
Returns the current chat session data for the chat visitor. This request is used to restore the session data for a chat visitor’s chat session after a ResyncSession request is sent.

The ChasitorSessionData request is the first message sent after a ResyncSession request is delivered.
Note: No messages should be sent after a 503 status code is encountered until this message is processed.

Syntax
Available since release
This resource is available in API versions 29.0 and later.

Response body
ChasitorSessionData response

SEE ALSO:
ResyncSession
Status Codes and Error Responses
Status Codes and Error Responses

ChatEnded
Indicates that the chat has ended.

Syntax
Available since release
This resource is available in API versions 29.0 and later.

Request Body
ChatEndReason—Include the ChatEndReason parameter in the request body of your request to specify the reason that the chat ended. For example: {reason: “client”}.

Response body
None

Response properties
None

ChatEstablished
Indicates that an agent has accepted a chat request and is engaged in a chat with a visitor.

Syntax
Available since release
This resource is available in API versions 29.0 and later.

Response body
ChatEstablished response

SEE ALSO:
ChatRequestSuccess
ChatMessage
Indicates a new chat message has been sent from an agent to a chat visitor.

Syntax
Available since release
This resource is available in API versions 29.0 and later.

Response body
ChatMessage response

ChatRequestFail
Indicates that the chat request was not successful.

Syntax
Available since release
This resource is available in API versions 29.0 and later.

Response body
ChatRequestFail response

ChatRequestSuccess
Indicates that the chat request was successful and routed to available agents.

Note: The ChatRequestSuccess response only indicates that a request has been routed to available agents. The chat hasn’t been accepted until the ChatEstablished response is received.

Syntax
Available since release
This resource is available in API versions 29.0 and later.

Response body
ChatRequestSuccess response

SEE ALSO:
ChatEstablished

ChatTransferred
Indicates the chat was transferred from one agent to another.

Syntax
Available since release
This resource is available in API versions 29.0 and later.

Response body
ChatTransferred response
**CustomEvent**

Indicates a custom event was sent from an agent to a chat visitor during a chat.

**Syntax**

*Available since release*

This resource is available in API versions 29.0 and later.

**Response body**

`CustomEvent response`

---

**NewVisitorBreadcrumb**

Indicates the URL of the Web page the chat visitor is currently viewing.

**Syntax**

*Available since release*

This resource is available in API versions 29.0 and later.

**Response body**

`NewVisitorBreadcrumb response`

---

**QueueUpdate**

Indicates the new position of the chat visitor in the chat queue when the visitor’s position in the queue changes.

**Syntax**

*Available since release*

This resource is available in API versions 29.0 and later.

**Response body**

`QueueUpdate response`

---

**MultiNoun**

Batches multiple POST requests together if you’re sending multiple messages at the same time.

**Syntax**

**URI**

`https://hostname/chat/rest/System/MultiNoun`

*Available since release*

This resource is available in API versions 29.0 and later.

**Formats**

JSON

**HTTP methods**

POST
**Customize the Chat Visitors’ Experience**

With the Live Agent visitor REST API resources, you can establish your chat visitors’ experience with Live Agent in custom mobile applications.

**IN THIS SECTION:**
- **Settings**
  - Retrieves all settings information about the Live Agent deployment that’s associated with your chat session. The Settings request is required as the first request to establish a chat visitor’s session.
- **Availability**
  - Indicates whether a chat button is available to receive new chat requests.
- **Breadcrumb**
  - Sets a breadcrumb value to the URL of the Web page that the chat visitor is viewing as the visitor chats with an agent. The agent can then see the value of the breadcrumb to determine the page the chat visitor is viewing.
- **VisitorId**
  - Generates a unique ID to track a chat visitor when they initiate a chat request and tracks the visitor’s activities as the visitor navigates from one Web page to another.

**Settings**

Retrieves all settings information about the Live Agent deployment that’s associated with your chat session. The Settings request is required as the first request to establish a chat visitor’s session.

**Syntax**

**URI**

https://hostname/chat/rest/Visitor/Settings
Available since release
This resource is available in API versions 29.0 and later.

Formats
JSON

HTTP methods
GET

Request headers
X-LIVEAGENT-API-VERSION

Request parameters
None

Query parameters

org_id
The ID of the Salesforce organization that’s associated with the Live Agent deployment

deployment_id
The ID of the Live Agent deployment that the chat request was initiated from

Settings.buttonIds
An array of chat button IDs for which to retrieve settings information

Settings.updateBreadcrumb
Indicates whether to update the chat visitor’s location with the URL of the Web page that the visitor is viewing

Request body
None

Response body
Settings response on page 36

Availability
Indicates whether a chat button is available to receive new chat requests.

Syntax

URI
https://hostname/chat/rest/Visitor/Availability

Available since release
This resource is available in API versions 29.0 and later.

Formats
JSON

HTTP methods
GET

Request headers
X-LIVEAGENT-API-VERSION

Request parameters
None
Query parameters

org_id
The ID of the Salesforce organization that’s associated with the Live Agent deployment

deployment_id
The ID of the Live Agent deployment that the chat request was initiated from

Availability.ids
An array of object IDs for which to verify availability

Request body
None

Response body
Availability response

Breadcrumb
Sets a breadcrumb value to the URL of the Web page that the chat visitor is viewing as the visitor chats with an agent. The agent can then see the value of the breadcrumb to determine the page the chat visitor is viewing.

Syntax

URI
https://hostname/chat/rest/Visitor/Breadcrumb

Available since release
This resource is available in API versions 29.0 and later.

Formats
JSON

HTTP methods
POST

Request headers
X-LIVEAGENT-API-VERSION

Request parameters
None

Query parameters
None

Request body
Breadcrumb request

Response body
None

VisitorId
Generates a unique ID to track a chat visitor when they initiate a chat request and tracks the visitor’s activities as the visitor navigates from one Web page to another.
Syntax

URI
https://hostname/chat/rest/Visitor/VisitorId

Available since release
This resource is available in API versions 29.0 and later.

Formats
JSON

HTTP methods
GET

Request headers
X-LIVEAGENT-API-VERSION

Request parameters
None

Query parameters
- **org_id**
  The Salesforce organization ID
- **deployment_id**
  The ID of the Live Agent deployment that the chat request was initiated from

Request body
None

Response body
VisitorId response
To perform a POST or GET request, pass query parameters or create a request body that’s formatted in JSON. Request bodies can contain one or more other request bodies that are nested inside. Each request body can contain unique request properties.

Breadcrumbs

Request properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>location</td>
<td>String</td>
<td>The URL of the Web page that the chat visitor is viewing.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

Request body

```
"location":{
  "type":"string",
  "description":"The current location or URL of the visitor",
  "required":true,
  "version":29.0
}
```

ChasitorInit

Request properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>organizationId</td>
<td>String</td>
<td>The chat visitor’s Salesforce organization ID.</td>
<td>29.0</td>
</tr>
<tr>
<td>deploymentId</td>
<td>String</td>
<td>The ID of the deployment from which the chat originated.</td>
<td>29.0</td>
</tr>
<tr>
<td>buttonId</td>
<td>String</td>
<td>The ID of the button from which the chat originated.</td>
<td>29.0</td>
</tr>
<tr>
<td>sessionId</td>
<td>String</td>
<td>The chat visitor’s Live Agent session ID.</td>
<td>29.0</td>
</tr>
<tr>
<td>userAgent</td>
<td>String</td>
<td>The chat visitor’s browser user agent.</td>
<td>29.0</td>
</tr>
<tr>
<td>language</td>
<td>String</td>
<td>The chat visitor’s spoken language.</td>
<td>29.0</td>
</tr>
<tr>
<td>screenResolution</td>
<td>String</td>
<td>The resolution of the chat visitor’s computer screen.</td>
<td>29.0</td>
</tr>
</tbody>
</table>
### Request Bodies for Live Agent REST API

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>visitorName</td>
<td>String</td>
<td>The chat visitor’s custom name.</td>
<td>29.0</td>
</tr>
<tr>
<td>prechatDetails</td>
<td>Array of CustomDetail objects</td>
<td>The pre-chat information that was provided by the chat visitor.</td>
<td>29.0</td>
</tr>
<tr>
<td>prechatEntities</td>
<td>Array of Entity objects</td>
<td>The records to create when a chat visitor begins a chat.</td>
<td>29.0</td>
</tr>
<tr>
<td>buttonOverrides</td>
<td>Array of Strings</td>
<td>The button override rules that indicate how a chat request will be rerouted.</td>
<td>29.0</td>
</tr>
<tr>
<td>receiveQueueUpdates</td>
<td>Boolean</td>
<td>Indicates whether the chat visitor will receive queue position updates (true) or not (false).</td>
<td>29.0</td>
</tr>
<tr>
<td>isPost</td>
<td>Boolean</td>
<td>Indicates whether the chat request was made properly through a POST request (true) or not (false).</td>
<td>29.0</td>
</tr>
</tbody>
</table>

### Request body

```
{
    organizationId: "00DD0000000JVXs",
    deploymentId: "572D000000000J6",
    buttonId: "573D000000000OC",
    agentId: "005B0000000F3b2",
    doFallback: true,
    sessionId: "5503f854-0203-4324-8ed5-f793a367426f",
    userAgent: "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_6_8) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/28.0.1500.95 Safari/537.36",
    language: "en-US",
    screenResolution: "2560x1440",
    visitorName: "Jon A.",
    prechatDetails: [
        {
            label: "E-mail Address",
            value: "jon@example.com",
            transcriptFields: [
                "c__EmailAddress"
            ],
            displayToAgent: true
        }
    ],
    buttonOverrides: [
        "573D000000000OD"
    ],
    receiveQueueUpdates: true,
    isPost: true
}
```
ChasitorResyncState

**Request properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>organizationId</td>
<td>String</td>
<td>The chat visitor’s Salesforce organization ID.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

**Request body**

```json
{
  organizationId: "00DD0000000JVXs"
}
```

ChasitorSneakPeek

**Request properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>position</td>
<td>integer</td>
<td>The position of the Sneak Peek update in the chat.</td>
<td>29.0</td>
</tr>
<tr>
<td>text</td>
<td>String</td>
<td>The text that the chat visitor is typing in the text input area of the chat window.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

**Request body**

```json
{
  position: 3,
  text: "Hi there."
}
```

ChatMessage

**Request properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>The text of the chat visitor’s message to the agent.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

**Request body**

```json
{
  text: "I have a question about my account."
}
```
**CustomEvent**

Request properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>The type of custom event that occurred, used for adding the event listener on the agent's side.</td>
<td>29.0</td>
</tr>
<tr>
<td>data</td>
<td>String</td>
<td>Data that's relevant to the event that was sent to the agent.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

Request body

```json
{
    type: "PromptForCreditCard",
    data: "Visa"
}
```

**MultiNoun**

Request properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>nouns</td>
<td>Array of NounWrapper objects</td>
<td>An array of noun objects and their properties that are batched in the MultiNoun request.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

Request body

```json
{
    nouns: [
        {
            prefix: "Chasitor",
            noun: "ChatMessage",
            object: {
                text: "Goodbye"
            }
        },
        {
            prefix: "Chasitor",
            noun: "ChatEnd",
            object: {}
        }
    ]
}
```
A request to a Live Agent REST API resource returns a response code. The successful execution of a resource request can also return a response body in JSON format.

## Availability

### Response Properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Array of Result objects</td>
<td>A list of Salesforce IDs that correspond to agents and chat buttons and their respective availability to receive new chat requests.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

### Response body

```json
{
    "results":{
        "type": "array",
        "description": "List of ids and their availability",
        "items":{
            "name": "result",
            "type": "object",
            "properties":{
                "id":{
                    "type": "string",
                    "description": "The id of the entity",
                    "required": true,
                    "version": 29.0
                },
                "isAvailable":{
                    "type": "boolean",
                    "description": "Whether or not the entity is available for chat",
                    "required": true,
                    "version": 29.0
                }
            }
        },
        "required": true,
        "version": 29.0
    }
}
```
ChasitorSessionData

Response Properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>queuePosition</td>
<td>integer</td>
<td>The position of the chat visitor in the chat queue.</td>
<td>29.0</td>
</tr>
<tr>
<td>geoLocation</td>
<td>GeoLocation object</td>
<td>The chat visitor’s location, based on the IP address from which the request originated.</td>
<td>29.0</td>
</tr>
<tr>
<td>url</td>
<td>String</td>
<td>The URL that the chat visitor is visiting.</td>
<td>29.0</td>
</tr>
<tr>
<td>oref</td>
<td>String</td>
<td>The original URL that the chat request came from.</td>
<td>29.0</td>
</tr>
<tr>
<td>postChatUrl</td>
<td>String</td>
<td>The URL to which to redirect the chat visitor after the chat has ended.</td>
<td>29.0</td>
</tr>
<tr>
<td>sneakPeekEnabled</td>
<td>Boolean</td>
<td>Whether Sneak Peek is enabled for the agent who accepts the chat.</td>
<td>29.0</td>
</tr>
<tr>
<td>chatMessages</td>
<td>Array of TranscriptEntry objects</td>
<td>The chat message structure that’s synchronized across the agent.js and chasitor.js files.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

Response body

```json
{
    queuePosition: 1,
    geoLocation: {
        countryCode: "US",
        countryName: "United States of America",
        region: "CA",
        city: "San Francisco",
        organization: "Salesforce.com",
        latitude: 37.793880,
        longitude: -122.395114
    },
    url: "http://yoursite",
    oref: "http://www.google.com?q=yoursite",
    postChatUrl: "http://yoursite/postchat",
    sneakPeekEnabled: true,
    chatMessages: [{
        type: "Agent",
        name: "Andy L.",
        content: "Hello, how can I help you?"
    }]
}
```
Response Bodies for Live Agent REST API

ChasitorIdleTimeoutWarningEvent

Response properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>idleTimeoutWarningEvent</td>
<td>String</td>
<td>Informs the server when a warning is triggered or cleared. Possible values: triggered and cleared.</td>
<td>35.0</td>
</tr>
</tbody>
</table>

ChatEstablished

Response properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the agent who is engaged in the chat.</td>
<td>29.0</td>
</tr>
<tr>
<td>userId</td>
<td>String</td>
<td>The user ID of the agent who is engaged in the chat.</td>
<td>29.0</td>
</tr>
<tr>
<td>sneakPeekEnabled</td>
<td>Boolean</td>
<td>Whether Sneak Peek is enabled for the agent who accepts the chat.</td>
<td>29.0</td>
</tr>
<tr>
<td>chasitorIdleTimeout</td>
<td>ChasitorIdleTimeoutSettings</td>
<td>Gives the settings for chat visitor idle time-out.</td>
<td>35.0</td>
</tr>
</tbody>
</table>

Response body

```json
{
    name: "Andy L.",
    userId: "fidda237-57f8-4816-b8e8-59775f1e44c8",
    ```
Response Bodies for Live Agent REST API

### ChatMessage

**Response properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the agent who is engaged in the chat.</td>
<td>29.0</td>
</tr>
<tr>
<td>text</td>
<td>String</td>
<td>The text of the chat message that the agent sent to the chat visitor.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

**Response body**

```
{
    name: "Andy L."
    text: "Hello, how can I help you?"
}
```

### ChatRequestFail

**Response properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>reason</td>
<td>String</td>
<td>The reason why the chat request failed—for example, no agents were available to chat or an internal error occurred.</td>
<td>29.0</td>
</tr>
<tr>
<td>postChatUrl</td>
<td>String</td>
<td>The URL of the post-chat page to which to redirect the chat visitor after the chat has ended.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

**Response body**

```
{
    reason: "Unavailable",
    postChatUrl: "http://yoursite/postChat"
}
```
ChatRequestSuccess

Response properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>queuePosition</td>
<td>integer</td>
<td>The position of the chat visitor in the chat queue.</td>
<td>29.0</td>
</tr>
<tr>
<td>geoLocation</td>
<td>GeoLocation object</td>
<td>The chat visitor’s location, based on the IP address from which the request originated.</td>
<td>29.0</td>
</tr>
<tr>
<td>url</td>
<td>String</td>
<td>The URL that the chat visitor is visiting.</td>
<td>29.0</td>
</tr>
<tr>
<td>oref</td>
<td>String</td>
<td>The original URL that the chat request came from.</td>
<td>29.0</td>
</tr>
<tr>
<td>postChatUrl</td>
<td>String</td>
<td>The URL to which to redirect the chat visitor after the chat has ended.</td>
<td>29.0</td>
</tr>
<tr>
<td>customDetails</td>
<td>Array of CustomDetail objects</td>
<td>The custom details of the deployment from which the chat request was initiated.</td>
<td>29.0</td>
</tr>
<tr>
<td>visitorId</td>
<td>String</td>
<td>The ID of the chat visitor.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

Response body

```
{
  queuePosition: 1,
  geoLocation: {
    countryCode:"US",
    countryName: "United States of America",
    region: "CA",
    city: "San Francisco",
    organization: Salesforce.com,
    latitude: 37.793880,
    longitude: -122.395114
  },
  url: "http://yoursite",
  oref: "http://www.google.com?q=yoursite",
  postChatUrl: "http://yoursite/postchat",
  customDetails: [
    {
      label: "E-mail Address",
      value: "jon@example.com",
      transcriptFields: [
        "c__EmailAddress"
      ],
      displayToAgent: true
    }
  ]
}
```
ChatTransferred

Response properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the agent to whom the chat was transferred.</td>
<td>29.0</td>
</tr>
<tr>
<td>userId</td>
<td>String</td>
<td>The ID of the chat visitor.</td>
<td>29.0</td>
</tr>
<tr>
<td>sneakPeekEnabled</td>
<td>Boolean</td>
<td>Whether Sneak Peek is enabled for the agent to whom the chat was transferred.</td>
<td>29.0</td>
</tr>
<tr>
<td>chasitorIdleTimeout</td>
<td>ChasitorIdleTimeoutSettings</td>
<td>Gives the settings for chat visitor idle time-out.</td>
<td>35.0</td>
</tr>
</tbody>
</table>

Response body

```json
{
  name: "Ryan S.",
  userId: "edacfa56-b203-43d5-9e1b-678278b61263",
  sneakPeekEnabled: false
}
```

CustomEvent

Response properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>The type of custom event that occurred, used for adding the event listener on the chat visitor's side.</td>
<td>29.0</td>
</tr>
<tr>
<td>data</td>
<td>String</td>
<td>Data that's relevant to the event that was sent to the chat visitor.</td>
<td>29.0</td>
</tr>
</tbody>
</table>
Response body

```
{
    type: "CreditCardEntered",
    data: "5105105105105100"
}
```

Messages

Response Properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>messages</td>
<td>Array of Message objects</td>
<td>The messages that was sent over the course of a chat.</td>
<td>29.0</td>
</tr>
<tr>
<td>sequence</td>
<td>integer</td>
<td>The sequence of the message as it was received over the course of a chat.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

Response body

```
{
    messages: [
        {
            type: "ChatEstablished",
            message: {
                name: "Andy L.",
                userId: "f1dda237-57f8-4816-b8e8-59775f1e44c8",
                sneakPeekEnabled: true
            }
        },
        {
            message: {
                name: "Andy L.",
                userId: "f1dda237-57f8-4816-b8e8-59775f1e44c8",
                sneakPeekEnabled: true
            }
        }
    ],
    sequence: 1
}
```

NewVisitorBreadcrumb

Response properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>location</td>
<td>String</td>
<td>The URL of the Web page that the chat visitor is viewing.</td>
<td>29.0</td>
</tr>
</tbody>
</table>
Response body

```json
{
  location: "http://yoursite/page2"
}
```

**QueueUpdate**

Response properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>position</td>
<td>integer</td>
<td>The updated position of the chat visitor in the chat queue.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

Response body

```json
{
  position: 3
}
```

**ResyncSession**

Response properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>isValid</td>
<td>Boolean</td>
<td>Indicates whether the session is valid (true) or not (false).</td>
<td>29.0</td>
</tr>
<tr>
<td>key</td>
<td>String</td>
<td>The session key for the new session after the old session has been reestablished.</td>
<td>29.0</td>
</tr>
<tr>
<td>affinityToken</td>
<td>String</td>
<td>The affinity token for the session that’s passed in the header for all future requests.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

Response body

```json
{
  isValid: true,
  key: "dd19849c-26df-4e21-b4cf-3aada410f35d",
  affinityToken: "73061fa0"
}
```
### SessionId

**Response Properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>The session ID for the new session.</td>
<td>29.0</td>
</tr>
<tr>
<td>key</td>
<td>String</td>
<td>The session key for the new session.</td>
<td>29.0</td>
</tr>
<tr>
<td>affinityToken</td>
<td>String</td>
<td>The affinity token for the session that's passed in the header for all future requests.</td>
<td>29.0</td>
</tr>
<tr>
<td>clientPollTimeout</td>
<td>integer</td>
<td>The number of seconds before you must make a Messages request before your Messages long polling loop times out and is terminated.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

**Response body**

```json
{
  "id": "241590f5-2e59-44b5-af89-9cae83bb6947",
  "key": "f6c1d699-84c7-473f-b194-abf4bf7cccf8!b65b13c7-f597-4dd2-aa3a-cbe01e69f19c",
  "affinityToken": "73061fa0",
  "clientPollTimeout": "30"
}
```

### Settings

**Response Properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>pingrate</td>
<td>number</td>
<td>The rate at which the visitor must ping the server to maintain the Live Agent visitor session.</td>
<td>29.0</td>
</tr>
<tr>
<td>contentServerUrl</td>
<td>String</td>
<td>The URL of the content server.</td>
<td>29.0</td>
</tr>
<tr>
<td>buttons</td>
<td>Array of button objects</td>
<td>A list of chat buttons, along with their settings information, that were specified when you made the Settings request.</td>
<td>29.0</td>
</tr>
</tbody>
</table>
Response body

```json
{
    "pingRate": {
        "type": "number",
        "description": "The rate at which the visitor should ping the server to maintain presence",
        "required": true,
        "version": 29.0
    },
    "contentServerUrl": {
        "type": "string",
        "description": "The content server URL",
        "required": true,
        "version": 29.0
    },
    "buttons": {
        "type": "array",
        "description": "The list of buttons",
        "items": {
            "name": "button",
            "type": "object",
            "properties": {
                "id": {
                    "type": "string",
                    "description": "The id of the button",
                    "required": true,
                    "version": 29.0
                },
                "type": {
                    "type": "string",
                    "description": "The type of the button",
                    "required": true,
                    "version": 29.0,
                    "enum": ["Standard", "Invite", "ToAgent"]
                },
                "endpointUrl": {
                    "type": "string",
                    "description": "The custom chat window url of the button",
                    "required": false,
                    "version": 29.0
                },
                "prechatUrl": {
                    "type": "string",
                    "description": "The prechat url of the button",
                    "required": false,
                    "version": 29.0
                },
                "language": {
                    "type": "string",
                    "description": "The language setting of the button",
                    "required": false,
                    "version": 29.0
                }
            }
        }
    }
}
```
"isAvailable":{
  "type":"boolean",
  "description":"Whether or not the button is available for chat",
  "required":true,
  "version":29.0
},
/* Invite related settings */
"inviteImageUrl":{
  "type":"string",
  "description":"The image of the button",
  "required":false,
  "version":29.0
},
"inviteImageWidth":{
  "type":"number",
  "description":"The width of the button image",
  "required":false,
  "version":29.0
},
"inviteImageHeight":{
  "type":"number",
  "description":"The height of the button image",
  "required":false,
  "version":29.0
},
"inviteRenderer":{
  "type":"string",
  "description":"The animation option of the invite",
  "required":false,
  "version":29.0,
  "enum": ["Slide", "Fade", "Appear", "Custom"]
},
"inviteStartPosition":{
  "type":"string",
  "description":"The start position of the animation",
  "required":false,
  "version":29.0,
  "enum": ["TopLeft", "TopLeftTop", "Top", "TopRightTop", "TopRight", "TopRightRight", "Right", "BottomRightRight", "BottomRight", "BottomRightBottom", "Bottom", "BottomLeftBottom", "BottomLeft", "BottomLeftLeft", "Left", "TopLeftLeft"]
},
"inviteEndPosition":{
  "type":"string",
  "description":"The end position of the animation",
  "required":false,
  "version":29.0,
  "enum": ["TopLeft", "Top", "TopRight", "Left", "Center", "Right", "BottomLeft", "Bottom", "BottomRight"]
}
}
VisitorId

Response Properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>sessionId</td>
<td>String</td>
<td>The session ID for the new session.</td>
<td>29.0</td>
</tr>
</tbody>
</table>

Response body

"sessionId":{
  "type":"string",
  "description":"The session id of the visitor",
  "required":true,
}
Response Bodies for Live Agent REST API

```json
"version": 29.0
}
```

SEE ALSO:

- Status Codes and Error Responses
A request to a Live Agent REST API resource returns a response code. The successful execution of a resource request can also return a response body in JSON format. Some response bodies return data types that contain their own properties. All property values that refer to a name of an entity or field are case-sensitive.

### Button

**Properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Required</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>The ID of the chat button object.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>type</td>
<td>String</td>
<td>The button type. Valid values are:</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Standard</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Invite</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ToAgent</td>
<td></td>
<td></td>
</tr>
<tr>
<td>endpointUrl</td>
<td>String</td>
<td>The URL of the custom chat window that's assigned to the chat button.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>prechatUrl</td>
<td>String</td>
<td>The URL of the pre-chat form that's assigned to the button.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>language</td>
<td>String</td>
<td>The chat button's default language.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>isAvailable</td>
<td>Boolean</td>
<td>Specifies whether the chat button is available to receive new chat requests (true) or not (false).</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>inviteImageUrl</td>
<td>String</td>
<td>The URL to the automated invitation's static image resource.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>inviteImageWidth</td>
<td>number</td>
<td>The width in pixels of the automated chat invitation's image.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
</tbody>
</table>
## Live Agent REST API Data Types

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Required</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>inviteImageHeight (for automated chat invitations only)</td>
<td>number</td>
<td>The height in pixels of the automated chat invitation’s image.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
</tbody>
</table>
| inviteRenderer (for automated chat invitations only) | String        | The animation option that’s assigned to the automated chat invitation. Valid values are:  
  - Slide  
  - Fade  
  - Appear  
  - Custom | FALSE    | 29.0               |
| inviteStartPosition (for automated chat invitations only) | String        | The position at which the automated chat invitation begins its animation. Valid values are:  
  - TopLeft  
  - TopLeftTop  
  - Top  
  - TopRightTop  
  - TopRight  
  - TopRightRight  
  - Right  
  - BottomRightRight  
  - BottomRight  
  - BottomRightBottom  
  - Bottom  
  - BottomLeftBottom  
  - BottomLeft  
  - BottomLeftLeft  
  - Left  
  - TopLeftLeft | FALSE    | 29.0               |
| inviteEndPosition (for automated chat invitations only) | String        | The position at which the automated chat invitation begins its animation. Valid values are:  
  - TopLeft | FALSE    | 29.0               |
### Live Agent REST API Data Types

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Required</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>hasInviteAfterAccept</td>
<td>Boolean</td>
<td>Specifies whether the automated chat invitation can be sent again after the customer accepted a previous chat invitation (true) or not (false).</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>hasInviteAfterReject</td>
<td>Boolean</td>
<td>Specifies whether the automated chat invitation can be sent again after the customer rejected a previous chat invitation (true) or not (false).</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>inviteRejectTime</td>
<td>number</td>
<td>The amount of time in seconds that the invitation will appear on a customer’s screen before the invitation is automatically rejected.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>inviteRules</td>
<td>Object</td>
<td>The custom rules that govern the behavior of the automated chat invitation, as defined in your custom Apex class.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
</tbody>
</table>

### CustomDetail

#### Properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Required</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>String</td>
<td>The customized label for the detail.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
</tbody>
</table>
### Property Names and Descriptions

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Required</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>The customized value for the detail.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>transcriptFields</td>
<td>Array of Strings</td>
<td>The names of fields to which to save the customer's details on the chat transcript.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>displayToAgent</td>
<td>Boolean</td>
<td>Specifies whether to display the customized detail to the agent (true) or not (false).</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
</tbody>
</table>

### Entity Properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Required</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>entityName</td>
<td>String</td>
<td>The record to search for or create.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>showOnCreate</td>
<td>Boolean</td>
<td>Specifies whether to display the record after it’s created (true) or not (false).</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>linkToEntityName</td>
<td>String</td>
<td>The name of the record to which to link the detail.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>linkToEntityField</td>
<td>String</td>
<td>The field within the record to which to link the detail.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>saveToTranscript</td>
<td>String</td>
<td>The name of the transcript field to which to save the record.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>entityFieldsMaps</td>
<td>Array of EntityFieldsMaps objects</td>
<td>The fields to which to associate the detail on a record.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
</tbody>
</table>
## EntityFieldsMaps

**Properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Required</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The name of the field to which to associate the detail.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>label</td>
<td>String</td>
<td>The customized label for the detail.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>doFind</td>
<td>Boolean</td>
<td>Specifies whether to use the field fieldName to perform a search for matching records (true) or not (false).</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>isExactMatch</td>
<td>Boolean</td>
<td>Specifies whether to only search for records that have fields that exactly match the field fieldName (true) or not (false).</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>doCreate</td>
<td>Boolean</td>
<td>Specifies whether to create a record based on the field fieldName if one doesn’t exist (true) or not (false).</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
</tbody>
</table>

## GeoLocation

**Properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Required</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>countryCode</td>
<td>String</td>
<td>The ISO 3166-1 alpha-2 country code for the chat visitor’s location.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>countryName</td>
<td>String</td>
<td>The name of the country that’s associated with the chat visitor’s location.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>region</td>
<td>String</td>
<td>The principal administrative division associated with the chat</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
</tbody>
</table>
### Live Agent REST API Data Types

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Required</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>visitor’s location</td>
<td>—</td>
<td>for example, the state or province.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>city</td>
<td>String</td>
<td>The name of the city associated with the chat visitor’s location.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>organization</td>
<td>String</td>
<td>The name of the organization associated with the chat visitor’s location.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>latitude</td>
<td>number</td>
<td>The latitude associated with the chat visitor’s location.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
<tr>
<td>longitude</td>
<td>number</td>
<td>The longitude associated with the chat visitor’s location.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
</tbody>
</table>

### Message

**Properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Required</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>The type of message that was received.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>message</td>
<td>Object</td>
<td>A placeholder object for the message that was received. Can return any of the responses that are available for the Messages request.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
</tbody>
</table>

### NounWrapper

**Properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Required</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>prefix</td>
<td>String</td>
<td>The prefix of the resource.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>noun</td>
<td>String</td>
<td>The name of the resource.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>data</td>
<td>String</td>
<td>The data to post to the resource.</td>
<td>FALSE</td>
<td>29.0</td>
</tr>
</tbody>
</table>
### Result

**Properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Required</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>The Salesforce ID of the agent or chat button.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>isAvailable</td>
<td>Boolean</td>
<td>Indicates whether the entity that’s associated with the Salesforce ID id is available to receive new chat requests (true) or not (false).</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
</tbody>
</table>

### TranscriptEntry

**Properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Required</th>
<th>Available Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Enumeration of type String</td>
<td>The type of message in the chat transcript. Valid values are:</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Agent: a message from an agent to a chat visitor</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Chatsitor: a message from a chat visitor to an agent</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- OperatorTransferred: A request to transfer a chat to another agent</td>
<td></td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the person who sent the chat message.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>content</td>
<td>String</td>
<td>The body of the message.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>timestamp</td>
<td>number</td>
<td>The date and time when the message was sent.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
<tr>
<td>sequence</td>
<td>number</td>
<td>The sequence in which the message was received in the chat.</td>
<td>TRUE</td>
<td>29.0</td>
</tr>
</tbody>
</table>

**SEE ALSO:**

[Status Codes and Error Responses](#)
Each request returns a status code or error response to indicate whether or not the request was successful.

When an error occurs or when a response is successful, the response header contains an HTTP code, and the response body usually contains:

- The HTTP response code
- The message accompanying the HTTP response code

<table>
<thead>
<tr>
<th>HTTP response code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>“OK” success code.</td>
</tr>
<tr>
<td>202</td>
<td>“Accepted” success code, for POST request.</td>
</tr>
<tr>
<td>204</td>
<td>“No Content” success code for Message request; you should resend the request as part of the message loop.</td>
</tr>
<tr>
<td>400</td>
<td>The request couldn’t be understood, usually because the JSON body contains an error.</td>
</tr>
<tr>
<td>403</td>
<td>The request has been refused because the session isn’t valid.</td>
</tr>
<tr>
<td>404</td>
<td>The requested resource couldn’t be found. Check the URI for errors.</td>
</tr>
<tr>
<td>405</td>
<td>The method specified in the Request-Line isn’t allowed for the resource specified in the URI.</td>
</tr>
<tr>
<td>500</td>
<td>An error has occurred within the Live Agent server, so the request couldn’t be completed. Contact Customer Support.</td>
</tr>
<tr>
<td>503</td>
<td>The affinity token has changed. You must make a ResyncSession request to get a new affinity token and session key, then make a ChasitorSessionData request to reestablish the chat visitor’s data within the new session.</td>
</tr>
</tbody>
</table>

SEE ALSO:

- Your Message Long Polling Loop
- ResyncSession
- ChasitorSessionData
- Response Bodies for Live Agent REST API
- Live Agent REST API Data Types
- ResyncSession
- ChasitorSessionData
INDEX

A
AgentDisconnect 16
AgentNotTyping 16
AgentTyping 16
API Endpoint 2
Availability 21

B
Breadcrumb 22

C
ChasitorInit 6
ChasitorNotTyping 9
ChasitorResyncState 8
ChasitorSessionData 16
ChasitorSneakPeek 10
ChasitorTyping 10
ChatEnd 11
ChatEnded 17
ChatEstablished 17
ChatMessage 12, 18
ChatRequestFail 18
ChatRequestSuccess 18
ChatTransferred 18
CustomEvent 12, 19

D
Data Types 41

E
Error responses 48

M
Messages 13
MultiNoun 19

N
NewVisitorBreadcrumb 19

Q
QueueUpdate 19

R
Requests
  Headers 3

Resources
  AgentDisconnect 16
  AgentNotTyping 16
  AgentTyping 16
  API Endpoint 2
  Availability 21
  Breadcrumb 22
  ChasitorInit 6
  ChasitorNotTyping 9
  ChasitorResyncState 8
  ChasitorSessionData 16
  ChasitorSneakPeek 10
  ChasitorTyping 10
  ChatEnd 11
  ChatEnded 17
  ChatEstablished 17
  ChatMessage 12, 18
  ChatRequestFail 18
  ChatRequestSuccess 18
  ChatTransferred 18
  CustomEvent 12, 19
  Data Types 41
  Long Polling 4
  Messages 13–14
  MultiNoun 19
  NewVisitorBreadcrumb 19
  QueueUpdate 19
  Request headers 3
  Requests 2, 5–6, 9, 20, 24
  Responses 28
  ResyncSession 7
  SessionId 5
  Settings 20
  VisitorId 22

Status codes 48

VisitorId 22